

---

# **click-shell Documentation**

***Release 2.2.dev0***

**Clark Perkins**

**Jun 27, 2021**



---

## Contents

---

<b>1</b>	<b>Features</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Usage . . . . .	3
1.3	Changelog . . . . .	5
1.4	Troubleshooting . . . . .	5



click-shell is an extension to [click](#) that easily turns your click app into a shell utility. It is built on top of the built in python [cmd](#) module, with modifications to make it work with click.

click-shell is compatible with python versions 2.7, 3.5, 3.6, 3.7, and 3.8.



# CHAPTER 1

---

## Features

---

- Adds a “shell” mode **with command completion** to any click app
- Just a one line change for most click apps

---

**Note:** It should be noted that click-shell **only** alters functionality if no arguments are passed on the command line. Previously if no arguments were passed, the help was displayed.

---

## 1.1 Installation

The easiest way to install is with pip:

```
pip install click-shell
```

If you'd rather, you can clone the github repo and install manually:

```
git clone https://github.com/clarkperkins/click-shell.git
python setup.py install
```

## 1.2 Usage

There are 2 main ways to utilize click-shell: the decorator and the factory method.

### 1.2.1 Decorator

The easiest way to get going with click-shell is with the click style decorator. `@click_shell.shell` is meant to replace click's `@click.group` decorator for the root level of your app. In fact, the object generated

by `@click_shell.shell` is a `click_shell.core.Shell` object, which is a subclass of `click.core.Group`.

```
from click_shell import shell

# @click.group() # no longer
@shell(prompt='my-app > ', intro='Starting my app...')
def my_app():
    pass

@my_app.command()
def testcommand():
    print('testcommand is running')

# more commands

if __name__ == '__main__':
    my_app()
```

When run with the above arguments, you should expect an output like so:

```
$ python my_app.py
Starting my app...
my-app > testcommand
testcommand is running
my-app >
```

`@shell` takes 4 arguments:

- `prompt` - this will get printed as the beginning of each line in the shell. This can take a callable that will be called each time a prompt is printed. On Python 3 ONLY, if the callable takes an argument named `ctx`, the click context will be passed in as that argument. Defaults to `'(Cmd) '`
- `intro` - this will get printed once when the shell first starts Defaults to `None`, meaning nothing gets printed
- `hist_file` - this is the location of the history file used by the shell. Defaults to `'~/ .click-history'`
- `on_finished` - a callable that will be called when the shell exits. You can use it to clean up any resources that may need cleaning up.

`@shell` also takes arbitrary keyword arguments, and they are passed on directly to the constructor for the `click_shell.Shell` class.

## 1.2.2 Factory Method

If you'd rather not use decorators (or can't for some reason), you can manually create a shell object and start it up:

```
import click
from click_shell import make_click_shell

@click.group()
@click.pass_context
def my_app(ctx):
    pass

# Somewhere else in your code (as long as you have access to the root level Context_
object)
```

(continues on next page)



(continued from previous page)

```
shell = make_click_shell(ctx, prompt='my-app > ', intro='Starting my app...')
shell.cmdloop()
```

The first argument passed to `make_click_shell` must be the root level context object for your click application. The other 3 args (`prompt`, `intro`, `hist_file`) are the same as described above under the Decorator section.

## 1.3 Changelog

The changelog is located in GitHub:

<https://github.com/clarkperkins/click-shell/blob/master/CHANGELOG.rst>

## 1.4 Troubleshooting

### 1.4.1 Autocomplete

If autocomplete isn't working after installation, you may be missing the `readline` module. Try one of the following depending on your platform:

For macOS / linux (the `readline` extra):

```
pip install click-shell[readline]
```

For Windows / cygwin (the `windows` extra):

```
pip install click-shell[windows]
```